

EV316936472

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION FOR LETTERS PATENT

Network Connection Agents and Troubleshooters

Inventors:

Ya-Bing Chu

Dinarte R. Morais

Nataly Pogrebinski

ATTORNEY'S DOCKET NO. MS1-1500US

TECHNICAL FIELD

This subject matter disclosed herein relates generally to network connectivity and more specifically to network connection agents and troubleshooters.

BACKGROUND

As more and more home and office electronic devices (e.g., smart devices, entertainment electronics, etc.) join the ranks of consumer products that can couple to and communicate over a network, a need has arisen for protocols and methods to establish correct network connections and settings.

The task of connecting to a network can daunt the skilled and overwhelm the uninitiated. User manuals and installation programs that include directions for connecting a device, such as a modem, game, etc., to the Internet disseminate cryptic and poorly translated instructions fraught with programming code segments and written for a technician with a knowledge of the device commensurate with the manufacturer's.

Known software, e.g., MOTIVE XI SMARTSERVICE SUITE, can automate a pre-installation interaction between a broadband provider and a customer, but this is generally for provisioning cable and digital subscriber line (DSL) networks at the broadband multi-system operator (MSO) level. (Motive Communications, Austin, Texas.)

A successful connection to the Internet or an online service on the first try often results from good luck rather than from an understanding of and an adherence to installation instructions. Unsuccessful attempts at connection are met with unhelpful recorded messages at the customer service number. An

1 unsuccessful user may not be able to obtain relevant help, as it may not be clear
2 whether a problem exists on the network or closer to home as when a mandatory
3 cable, driver, or wireless component is missing from an installation kit.
4
5

6 SUMMARY

7 Subject matter includes a network connection agent and troubleshooter that
8 automatically connects a device to a network service in multiple stages, displays
9 real-time status reports of connection progress with respect to each stage, and
10 displays troubleshooting help and instructions with respect to a stage if the stage is
11 not successful. The network connection agent and troubleshooter may use
12 multiple techniques for each stage, and if one technique is unsuccessful, another
13 technique is automatically attempted. Manual entry of network settings can be
14 selected during automatic connection.

15 In one implementation, a network connection agent and troubleshooter
16 verifies communicative coupling between a device and a network, detects IP
17 settings using DHCP or PPPoE, performs a DNS name resolution, and confirms
18 settings by communicating with the network service being connect to.
19
20
21
22
23
24
25

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a block diagram of an exemplary computing environment in which the subject matter may be practiced.

Fig. 2 is a block diagram of an exemplary network connection-and-troubleshooting engine.

Fig. 3 is a flow diagram of an exemplary method of connecting a device to a network or network service.

Fig. 4 is a flow diagram of another exemplary method of connecting a device to a network or network service.

Fig. 5 is a graphic representation of exemplary user interfaces of an exemplary network connection agent and troubleshooter.

Fig. 6 is a graphic representation of an exemplary network status screen user interface.

Fig. 7 is a graphic representation of an exemplary help and troubleshooting screen user interface of an exemplary NCAT.

Fig. 8 is a block diagram of an exemplary computing device suitable for use with the subject matter.

DETAILED DESCRIPTION

Overview

The subject matter presented herein includes an exemplary network connection agent and troubleshooter (hereinafter “NCAT”) that communicatively couples an electronic device to a network and/or online network service (“service”) and troubleshoots connection failures, particularly initial attempts to plug-in and/or connect with the Internet or a service. An NCAT not only makes the various stages of a network connection set-up process more automatic, but offers smart messaging for updating a user—in real-time—of the status of each stage in a connection attempt and offers context-sensitive troubleshooting help. Thus, for a user, an exemplary NCAT improves the experience of installing a device as the user can see an automatic connection process in real-time, how much of the process is left to complete, the stage of the process at which an error occurs, and context-sensitive troubleshooting help if needed. From the standpoint of a vendor of a device (or an online service), an NCAT may make an online service or a consumer electronic device, such as a modem, game console, set top box, personal media recorder, personal computer, personal digital assistant, wireless router, etc., that connects to the Internet, more user-friendly, since the online service or the device can “connect itself” to a network and/or service and is thus more attractive to consumers.

Usually, an NCAT is able to connect a device to a network and/or network service more automatically and effortlessly (for the user) than conventional methods, and in most circumstances a user does not have to actively participate in making a connection as the NCAT automatically progresses from verifying

1 communicative couplings, such as rudimentary communicative couplings, and
2 then establishes connectivity on higher layers of TCP/IP or other network
3 protocols.

4 When trouble arises during a connection attempt to a network or service, an
5 NCAT includes a toolbox of workarounds, so that the trouble is often overcome
6 and a network connection is still established in many circumstances.

7 When establishment of a network connection is unsuccessful, an NCAT
8 includes another toolbox of context-sensitive troubleshooting instructions and
9 error logs.

10 **Fig. 1** shows an exemplary NCAT 100 in the context of a computing
11 environment 102. An NCAT 100 communicatively couples a device 104 and a
12 network 106, over which a service 108 may be available. The NCAT 100 then
13 tries to automatically establish all connections and communications needed for the
14 device 104 to communicate properly with the network 106 and/or service 108.

15 In one implementation, an NCAT 100 is a separate software and/or
16 hardware entity that may be used with a variety of devices 104 to be connected. In
17 another implementation, such as that illustrated, an NCAT 100 is a software and/or
18 hardware entity that can be integrated into the design of a device 104, such as a
19 modem, a game console for online games, a set top box, a personal media
20 recorder, a personal computer, a portable computer, a personal digital assistant, a
21 wireless router, etc., that connects to a network 106, such as the Internet. In one
22 implementation, when installed as an integrated or at least “included” part of a
23 design for a given type of device 104, each device 104 of that type, for example
24 each game console, can boot to an NCAT 100 thus providing a common way to
25

1 handle connection failures, even if the connection failures happen for different
2 reasons.

3 At each step in a connection process, if an NCAT 100 fails at one
4 connection technique it may try additional techniques. For example, when
5 automatic detection of network settings by an NCAT 100 is not successful an
6 NCAT 100 may try another protocol and/or offer context-sensitive help to assist a
7 user in fixing the problem without tech support. The help may be offered on
8 several levels, for example, user-friendly help and more detailed and technical
9 information for technicians and support engineers. Overall, the measure of
10 comprehensive help that can be offered by an NCAT 100 can decrease the number
11 of support calls to a service 108 targeted for connection and/or to a user's Internet
12 Service Provider (ISP). If a support call is needed, it may be shorter because an
13 NCAT 100 returns extended information that can be relayed to support personnel.

14 In one implementation, as illustrated in Fig. 1, an NCAT 100 divides the
15 task of communicatively coupling a device 104 to a network 106 and/or a service
16 108 into four stages: a communicative coupling stage 110, a network settings
17 configuration stage 112, a name resolution stage 114 (e.g., via a domain name
18 system (DNS)), and a service connection stage 116, wherein the device 104
19 establishes communication with a service 108 that may be associated with the
20 device 104 (i.e., if the device 104 is a game console, the service 108 may be an
21 online gaming site). However, other ways of dividing the task of establishing a
22 successful communicative connection can be used.

23 The communicative coupling stage 110 involves establishing physical
24 communication means with a network 106, for example, via a cable 118, and/or a
25 wireless coupling 119, such as an infrared coupling and/or a wireless

1 communication coupling adhering to the Institute of Electronics Engineers (IEEE)
2 specifications 802.11, 802.11a, 802.11b, 802.11g, BLUETOOTH®, HOMERF
3 shared wireless access protocol (SWAP), Wi-Fi, etc. In other words, an NCAT
4 100 tries to determine if a user has connected equipment properly and established
5 a communicative coupling with a network 106. If a cable 118 and/or wireless
6 coupling 119 for access to a network 106 is unplugged or becomes inoperative, the
7 connection process stops and the user is informed that a cable 118 and/or wireless
8 coupling 119 is unplugged or defective. If a proper connection for the cable 118
9 and/or wireless coupling 119 is detected or re-established (i.e. the user plugs it in),
10 the overall connection and troubleshooting process resumes.

11 Once the communicative coupling stage 110 between a device 104 and an
12 access point to a target network 106 has been successfully detected by an NCAT
13 100, the NCAT 100 enters the network settings configuration stage 112. An
14 NCAT 100 attempts contact with a user's ISP 120, for example, an NCAT 100
15 could try to establish IP setting via either dynamic host configuration protocol
16 (DHCP) or via point to point protocol over Ethernet (PPPoE) that is generally
17 provided with a digital subscriber line: DSL. In one implementation, if an IP
18 address is obtained using DHCP, then the network settings configuration stage 112
19 is successful and an NCAT 100 can progress to the third connection stage. It
20 should be noted that the user can also enter manual settings, such as a static IP
21 address, and an NCAT 100 will automatically retrieve information from a domain
22 name service (DNS) to minimize data entry. If a user tries to manually set an IP
23 address, but a DHCP server is not found, an NCAT 100 may stop the network
24 settings configuration stage 112 and inform the user that an IP address was not
25

1 found, and offer help in the form of detailed information about possible causes of
2 failure, such as an IP conflict or loose cables.

3 In one implementation, if no IP address is obtained, and the user has not
4 entered a static IP address, an NCAT 100 continues the network settings
5 configuration stage 112 by attempting to detect PPPoE concentrators. If these are
6 discovered, the NCAT 100 tries to connect using PPPoE. Thus, an NCAT 100
7 tries to automatically detect the presence of a PPPoE server if the DHCP protocol
8 fails to obtain an IP address for the device 104. (If a computer using
9 MICROSOFT WINDOWS® XP does not obtain an IP address from a DHCP
10 server, the computing device automatically assigns an internal address of the
11 form (169.xxx.xxx.xxx), but this internal address assigned by WINDOWS® XP
12 does not allow the user to connect to the Internet, nor is there an attempt to find
13 other methods to connect the device.) (Microsoft Corporation, Redmond, WA)

14 If the automatic attempt at completing a network settings configuration
15 stage 112 using PPPoE fails, the NCAT 100 may ask the user to enter correct
16 PPPoE credentials, such as a correct username and password for gaining entry,
17 and/or to enter other information provided by the user's ISP 120. In one
18 implementation, once a user supplies credentials, an NCAT 100 may try to
19 connect again automatically, in case the user has entered incorrect information.

20 If the network settings configuration stage 112 is successful, an NCAT 100
21 progresses to the name resolution stage 114. Failure of the name resolution stage
22 114 may indicate a problem either with settings of the device 104 being connected
23 or with the user's ISP 120. During exchange with a DNS 122, if a valid DNS
24 server is found, then an NCAT 100 may progress to the service connection stage
25 116.

1 In the service connection stage 116, an NCAT 100 tries to connect to a
2 desired service 108, for example, to an online gaming service if the device 104
3 being connected and/or set-up is a game console. Including an exemplary service
4 connection stage 116 in an overall connection process provides confirmation of
5 network settings obtained in previous stages of a connection process.

6 In some implementations, an NCAT 100 performs an analysis of a quality
7 of service (QoS) promised by a service 108 or required and/or suggested for a
8 device 104. If an NCAT 100 fails to connect to a service 108 or, if in some
9 implementations QoS data is invalid or insufficient, the NCAT 100 may inform
10 the user that the service 108 is not responding.

11 In some implementations, an automatic QoS check and/or a network state
12 debug log provide extended information to the user, tech support, and even a
13 development team for a device 104. This information is recorded at the time of
14 the connection attempt, and may be provided to tech support or a development
15 team with respect to a connection status displayed on the screen to the user. QoS
16 information captured for debugging may provide, for example, the bandwidth, the
17 latency, the presence of a universal plug and play (UPnP) router, etc., during a
18 connection attempt and/or failure, as well as the stage and/or state of the network
19 connection at the time of failure.

20 In one implementation, in response to a connection failure, the serial
21 number of the device 104 is displayed to the user and/or stored for support
22 personnel to check if there is a problem with the device 104.
23
24
25

Exemplary NCAT Engine

Fig. 2 shows an exemplary network connection-and-troubleshooting engine “NCAT engine” 200, including components to perform the exemplary connection stages 110, 112, 114, 116. Accordingly, a communicative coupling engine 202, a network settings engine 204, a name resolution engine 206, and a service connection engine 208 are communicatively coupled with each other and with control logic 210 as illustrated. A user-interface engine 212, a help & troubleshooting engine 214, and an error logging engine 216 are also coupled with the other engines and the control logic 210 as illustrated. A mode selector 218 may also be included to switch between automatic setting detection and manual entry of network settings. In some implementations, the service connection engine 208 may have an associated QoS module 220. An exemplary device 104, such as a personal computer 109, suitable as an environment for practicing an exemplary NCAT engine 200 is shown in Fig. 8.

The communicative coupling engine 202, as mentioned, may check the connection status of a cable 118, such as an Ethernet cable, or the status of a wireless coupling 119. In one implementation this may be accomplished by detecting a data packet flow through a cable 118, but in an alternative implementation a communicative coupling engine 202 may be determined by other means, such as measuring a voltage, current, and/or capacitance, however small, associated with the cable 118; by sending out an electrical signal that elicits a response that can be measured to indicate a connected or unconnected cable 118; or by detecting a signal in a wireless coupling 119. A measurable electrical signal is present in many network setups if a piece of equipment is properly connected via a cable or wireless coupling to a hub. Of course, the hub may not be

1 connected to a functioning network, but at this stage (110), a communicative
2 coupling engine 202 is usually only concerned with a local communicative
3 coupling. The communicative coupling engine 202 informs the user-interface
4 engine 212 and the help and troubleshooting engine 214 to display either a success
5 or failure message for the user along with a context-sensitive description of the
6 success or failure and if needed, troubleshooting directions.

7 In one implementation the network settings engine 204 aims to establish a
8 working IP address, gateway, subnet mask, etc., for the device 104 and/or
9 connection, or at least obtain preliminary settings. Where the IP address and/or
10 settings are stored is dependent on the type of device 104. Thus, the network
11 settings engine 204, sometimes in conjunction with the name resolution engine
12 206, tries to obtain the local network topology, and settings for connecting with
13 the network 106.

14 Depending on the network 106 or Internet configuration in use, an NCAT
15 engine 200 may try to detect the presence of DHCP, but if that does not work, then
16 an NCAT engine 200 may send packets as a “discovery” broadcast to attempt
17 detection of other exemplary protocols that may be in use, such as PPPoE. In one
18 implementation, if a PPPoE concentrator and/or server is detected, the name
19 resolution engine 206 may begin the next stage in the connection process. The
20 network settings engine 204 also informs the user-interface engine 212 and the
21 help and troubleshooting engine 214 to display either a success or failure message
22 for the user along with a context-sensitive description of the success or failure and
23 if needed, troubleshooting directions. In one implementation, an NCAT 100 tries
24 to detect a home router that supports universal plug and play (UPnP) and opens a
25 correct port if one is detected.

1 In one implementation, the name resolution engine 206 queries a DNS
2 server for name resolution. Already in possession of an IP address, an NCAT 100
3 may need popular names and URL's to look up other machines on the Internet.
4 For example, if the device 104 is a game console, an NCAT 100 may need the
5 name of an online game website and/or live servers associated with the vendor of
6 the device 104.

7 A name resolution engine 206 can perform name resolution in two parts.
8 First, a DNS server address needs to be obtained. Second, the DNS server needs
9 to be looked up, i.e., a connection made for the name resolution. If the name
10 resolution engine 206 does not successfully perform either of these two parts, then
11 the name resolution stage 114 is deemed to be a connection failure.

12 The name resolution engine 206 as well as the network settings engine 204,
13 in conjunction with the mode selector 218, may allow a user to enter settings and
14 other information manually. For example, the user may enter a static IP address
15 and/or associated domain name manually. In one implementation, an NCAT
16 engine 200 usually assumes the user input is correct without verification.

17 If the DNS name resolution is successful, the name resolution engine 206
18 sends a domain name and/or other settings and IP addresses to the service
19 connection engine 208 for the next stage in the connection process. The name
20 resolution engine 206 also informs the user-interface engine 212 and the
21 troubleshooting help engine 214 to display either a success or failure message for
22 the user along with a context-sensitive description of the success or failure and if
23 needed, troubleshooting directions.

24 The service connection engine 208 makes an actual connection and/or
25 communication with a service 108. Theoretically, once an NCAT 100 has verified

1 a communicative coupling (such as a physical network connection), obtained
2 proper IP settings, and possesses a domain name for a service 108, the ability to
3 connect and communicate with a service 108 is a logical certainty. Oftentimes,
4 however, unforeseen conditions warrant an actual communication with a service
5 108 before the goal of overall connection between a device 104 and a service 108
6 can be declared a success. The actual connection with a service 108 verifies that
7 connection can be made with the proper handshake, that a part of the Internet is
8 not down, and that the preceding connection steps have really returned correct
9 settings.

10 If an exemplary NCAT engine 200 includes a QoS module 220, then
11 verification of QoS parameters may also be performed before connection between
12 a device 104 and a network 106 and/or service 108 is declared a success to the
13 user. QoS statistics and/or parameters gathered and/or analyzed by an NCAT
14 engine 200 include upload bandwidth, download bandwidth, number of data
15 packets dropped or lost, latency, jitter, etc...and other QoS parameters known in
16 the art of providing a guaranteed data transfer ability. In some implementations,
17 an ongoing recordation of QoS information is made during the stages of a
18 connection attempt. This allows the error logging engine 216 to supply a user or
19 support personnel with QoS parameters at the time one of the connections stages
20 failed. For example, if a connection attempt fails to contact a DNS server, perhaps
21 a QoS parameter at the time of failure shows that the upload bandwidth was
22 temporarily reduced to near zero and the packet loss rate was unusually high.

23 The QoS statistics and/or parameters gathered by a QoS module 220 of an
24 NCAT engine 200 are used to accomplish a successful connection, but may also
25 be sent to the user-interface engine 212 for ongoing display to a user. In some

1 implementations, the QoS information is included in dashboard information of a
2 device 104 while in alternative implementations the QoS information may be
3 available only to technical personnel, or those who know an access method
4 generally undocumented to most consumers, such as a service menu of the device
5 104.

6 In some implementations, the user-interface engine 212 provides messages
7 that are considered status reports, such as “DHCP not available” or “IP not
8 detected,” while in other implementations the troubleshooting-help engine 214
9 may provide such messages as preludes for directions to be displayed for
10 remedying the events reported in the messages. In still other alternative
11 implementations, such messages are error messages, and therefore provided by the
12 error logging engine 216.

13 The error logging engine 216 may trap, record, and return extended error
14 codes and messages in addition to regular connection status and failure-to-connect
15 messages and information. Examples of extended error information include
16 values for QoS parameters at the time of a connection failure, the number of
17 connection attempts and/or logon attempts during various stages of a connection
18 attempt, uplink and downlink times, serial number of the device 104, etc. All
19 these extended error messages and codes are useful to support personnel solving a
20 problem that the user cannot.

21 In one implementation, an exemplary error logging engine 216 stores error
22 events and a record of connection failures including the above-mentioned
23 extended error information (such as a snapshot of the QoS environment during a
24 failure). The failure record and related error information can be persisted, for
25 example on a hard disk, and uploaded later when a connection with a service 108

1 is actually successful. In one implementation, a failure record and related error
2 information are used for statistical treatment of the connection characteristics of a
3 device 104 over multiple instances of connection failure. In one implementation, a
4 failure record and/or related error information can be used to complete a cycle of a
5 Bayesian engine, which can refine connection solutions for future users of a
6 device 104 or for customer service and technology planning purposes.

7 In one implementation, a service 108 can obtain uploaded failure records
8 and error information from many connection failure occurrences of a given type or
9 model of device 104 use failure occurrence as a node for a probabilistic belief
10 network (e.g., a Bayes network) employed as a decision-making tool. A Bayesian
11 network thus constructed can provide intuitive inference for computing the
12 probability distributions of a set of variables in the Bayes network given evidence
13 of other related variables in the network. In a device 104 having multiple parts
14 (steps, states, and/or modules), an exemplary Bayesian network can describe
15 probabilistic relationships between the parts and/or the connection stages 110, 112,
16 114, 116 and make decisions about one or more parts and/or connection stages
17 using probabilistic inferences about the behavior, state, and/or input from the other
18 parts and/or connection stages. A Bayesian network thus constructed from
19 connection failure information stored by one or more exemplary error logging
20 engines 216 may be used to decide which factors are influencing connection
21 failure with respect to a given type or model of device 104.

22 Of course, an NCAT engine 200 may have a different configuration than
23 the exemplary NCAT engine 200 illustrated. The task of automatically initiating a
24 connection between device 104 and a network 106 and/or network service 108
25 may be divided in many ways, and variations in the engines and modules of an

1 NCAT engine 200 may reflect variations in the division of the connection task into
2 different stages or segments.

4 Exemplary Methods

5 **Fig. 3** shows an exemplary method 300 of communicatively coupling a
6 device 104 to a network 106 and/or network service 108. In the flow diagram, the
7 operations are summarized in individual blocks. The operations of the exemplary
8 method 300 may be performed in hardware and/or as machine-readable
9 instructions (software or firmware) that can be executed by a processor or a
10 device, such as a device that includes an exemplary NCAT engine 200.

11 At block 302, a task of connecting a device to a network and/or network
12 service is divided into stages.

13 At block 304, one of the stages is selected. The selection usually proceeds
14 in a logical order, e.g., a communicative coupling is completed before IP settings
15 can be established, etc.

16 At block 306, a technique is attempted for completing the selected stage.
17 The technique may be one of multiple techniques that will be attempted to
18 complete the stage.

19 At block 308, a status report is displayed in real-time. The status report
20 may be very specific, i.e., a progress indicator of percentage of the technique
21 completed, or the status report may be more general, i.e., an indicator of whether
22 the technique succeeded or failed. Yet again, the status report may just indicate
23 whether the stage being attempted was completely successfully or unsuccessfully.

1 At block 310, if the technique for a stage is successful, then the exemplary
2 method branches to block 312, but if the technique is not successful, then the
3 exemplary method 300 branches to block 314.

4 At block 312, after a successful technique for completing a stage of
5 connection, if there are more stages to complete, then the exemplary method 300
6 branches back to block 304 to select a subsequent stage of connection to attempt.
7 If there are no more stages to complete then at block 316 a success status is
8 displayed and at block 318 the exemplary method 300 ends.

9 At block 314, if there are more techniques to attempt for completing a stage
10 of the connecting, then the exemplary method 300 branches to block 306 to
11 attempt the next technique. If there are no more techniques to attempt, then at
12 block 320 a failure status is displayed, troubleshooting help is presented, and at
13 block 322 the exemplary method is suspended or ends.

14 Fig. 4 shows another exemplary method 400 of communicatively coupling
15 a device 104 to a network 106 and/or network service 108. In the flow diagram,
16 the operations are summarized in individual blocks. The operations of the
17 exemplary method 400 may be performed in hardware and/or as machine-readable
18 instructions (software or firmware) that can be executed by a processor or a
19 device, such as a device that includes an exemplary NCAT engine 200.

20 At block 402, a status of a communicative coupling between a device 104
21 and a network 106 is determined, e.g., an NCAT 100 determines if an Ethernet
22 cable is plugged in and/or operational. If the communicative coupling is
23 operational, then the exemplary method branches to blocks 404 and 406. At block
24 404, a successful communicative coupling is displayed and/or indicated. If the
25 communicative coupling is not operational, then the exemplary method 400

1 branches to block 408 where an unsuccessful communicative coupling is displayed
2 or indicated along with help, which may include specific context-sensitive
3 instructions for troubleshooting.

4 At block 406, the exemplary method 400 commences IP settings
5 configuration. The exemplary method 400 determines if DHCP services are
6 configured (or bootstrap protocol "BOOTP" if in use), i.e., the exemplary method
7 400 attempts to get an IP address "lease." If successful, the exemplary method
8 400 branches to blocks 410 and 412. At block 410, a successful IP settings
9 configuration is displayed and/or indicated to a user. At block 412, the exemplary
10 method 400 commences a next stage, DNS name resolution, as will be discussed
11 below.

12 If at block 406, the DHCP services were not configured, then the exemplary
13 method 400 branches to block 414, where the exemplary method 400 determines if
14 PPPoE is required. If PPPoE services are required, then the exemplary method
15 branches to block 416, where a PPPoE connection is attempted. If a PPPoE
16 connection is successful, then at block 410 an IP settings configuration success is
17 displayed or indicated and the exemplary method 400 proceeds to block 412. But
18 at block 414, if PPPoE services were not required to configure IP settings (and at
19 block 406 DHCP did not work either) then at block 418 the exemplary method 400
20 displays or indicates an IP setting configuration failure along with help, which
21 may include specific context-sensitive instructions for troubleshooting.

22 At block 412, the exemplary method 400 proceeds to query DNS for an IP
23 address, i.e., DNS name resolution. If an IP address is successfully obtained, the
24 exemplary method branches to blocks 420 and 422. At block 420, a successful
25 DNS name resolution is displayed and/or indicated. At block 422, the exemplary

1 method 400 commences a next stage, connection with a service 108, as will be
2 discussed below. If at block 412 the DNS name resolution is not successful, then
3 at block 424 a DNS name resolution failure message is displayed or indicated.

4 At block 422, communication with a service 108 is attempted. If at block
5 422 the communication with a service 108 is successful, the exemplary method
6 branches to block 426. If the attempted communication with the service 108 is
7 unsuccessful, then the exemplary method branches to block 428, where a service
8 connection failure message is displayed or indicated.

9 At block 426, the exemplary method 400 performs a QoS check. If the
10 QoS check is successful, for example, meets minimum standards or selected
11 values for guaranteed bandwidth, then the exemplary method branches to block
12 430 to display or indicate a service connection success message and to block 432
13 to suspend or end the exemplary method. If at block 426 the QoS check is not
14 successful, even though communication with the service 108 may have been
15 established at block 422, the exemplary method branches to block 428, where a
16 service connection failure message is displayed or indicated.

17 The exemplary method 400 described above is just one example of how an
18 NCAT 100 can perform a connection process between a device 104 and a network
19 106 and/or service 108. Other variations and additions are possible, for example,
20 different protocols in different number and processed in a different sequence may
21 be employed. Likewise, each part of the exemplary method 400 can be broken
22 down into smaller or separate parts, for example, the process at block 426,
23 checking QoS, could be a discreet stage having specific success and failure display
24 consequences, instead of being combined with the service connection stage 116 at
25 block 422 and its indicators of failure and success at blocks 426 and 430.

Exemplary User Interfaces

Fig. 5 shows exemplary user interfaces (UIs) 500 for an implementation of an NCAT 100. A main screen 502 may display a welcome message and offer a choice between automatic detection and/or connection and manual entry of network settings. If automatic detection and/or connection is selected, then previous and/or default setting may be cleared and the main screen 502 links to a connection status screen 504 wherein real-time statuses of auto-connection and auto-detection during a network settings discovery process are displayed. In some implementations, navigation between screens is suspended during auto-connection and auto-detection. However, a cancel link 506 may be provided to stop the automatic process. A first settings link 508 may also be provided to allow a user the option to break into the automatic process and add manual settings, for example at a settings entry screen (512—to be discussed below). The first settings link 508 may also be highlighted by an exemplary NCAT 100 when information is required from a user, for example, when an NCAT 100 is requesting PPPoE credentials from the user. A help link 510 may be included to request help even though an NCAT 100 automatically returns context-sensitive help and troubleshooting directions when a stage of connection fails.

If a user selects manual entry of network settings, then the main screen 502 links to a network settings input screen 512, where a user can enter settings such as IP information, DNS servers, host name(s), and PPPoE settings, etc. The settings input screen 512 may also include a “test connection” link 514, to perform a trial run of the settings entered by the user.

The “test connection” link 514 may invoke a network status screen 516 that displays auto-connection attempts in real-time at each exemplary connection stage

1 using the settings supplied by the user. If a failure occurs, then a second settings
2 link 518 may link the user back to the settings entry screen 512. A “done” link
3 520 elicits a user’s satisfaction with the settings entered and the status results
4 displayed.

5 **Fig. 6** shows the exemplary network connection status screen 504 of Fig. 5
6 in greater detail. The connection status screen 504 displays a status message for
7 each stage of a connection process. Although the illustrated connection status
8 screen 504 indicates four stages in the connection process, a different number of
9 stages could be used in a connection process giving rise to a corresponding
10 number of status messages in a connection status screen 504.

11 Typically, a network connection status screen 504 displays a different
12 message for a given stage depending on the status of the connection attempt for
13 that stage. For example, a communicative coupling stage 110 might have real-
14 time status messages that recite, “cables not connected,” “cable connection
15 underway,” or “cables connected.” The connection status of each stage of a
16 connection process can be shown graphically at the same time as, or separately
17 from, the display of the status messages.

18 In the illustrated example, a communicative coupling status message 602 is
19 displayed to indicate the real-time status of a communicative coupling stage 110
20 and a corresponding graphic indicator 610, in this case an image of a light or
21 button that is initially off or empty, is turned on or is filled in with a pattern or a
22 solid color to indicate a successful completion of the stage. If no attempt at
23 completing the stage has yet been made, then a graphic indicator (e.g., as in 610)
24 may be left turned off or blank, and if a stage has failed, an indicator may show a
25 failure signal, such as a large “X” placed over or through an empty graphic

indicator (e.g., as in 614). Likewise, an IP settings configuration status message 604 is displayed to indicate a real-time status of a network settings configuration stage 112 and a corresponding graphic indicator 612 that is initially empty is filled in with a pattern or a solid color to indicate a successful completion of the stage. A DNS name resolution status message 606 is likewise displayed to indicate the real-time status of a name resolution stage 114. In the illustrated example, the attempt has failed, so a “DNS name not resolved” status message 606 is displayed and a corresponding graphic indicator 614 is crossed-out with a large “X.” A service connection status message 608 is displayed to indicate the real-time status of a service connection stage 116 and in the illustrated example, since no attempt has been made to connect to the service, a corresponding graphic indicator 616 is left turned off or empty.

Fig. 7 shows an exemplary help screen 700 automatically generated by an NCAT 100. An exemplary help screen 700 uses smart messaging. The information provided in an exemplary help screen 700 is context-sensitive and automatically generated in response to a real-time connection status. For example, if during a name resolution stage 114 the DNS servers cannot be reached, an exemplary help screen 700 might contain several pieces of information, such as a general status message 702, “DNS error,” that informs the user of the stage in a connection process wherein the failure is occurring, and a more specific status message 704, “the DNS servers could not be reached,” that informs the user of the specific operation or technique within a stage of connecting that has failed. Such responsive real-time information is much more context-sensitive than conventional messages, such as “process aborted,” “error A751,” “connection failed,” “operation could not be completed,” or a spinning cursor that stops spinning.

1 An exemplary help screen 700 also provides context-sensitive
2 troubleshooting help 706, such as “verify that the DNS server addresses match the
3 settings provided by your ISP.” While this example message recites
4 troubleshooting help that most users would not want to receive (because a user
5 may not have received any settings from their ISP), still, the troubleshooting help
6 706 is context-sensitive and has already narrowed troubleshooting down to a
7 specific problem, that if addressed by a user, solves the failure or at least solves a
8 first step in a solution to the failure. Thus, the contextual status reports and
9 troubleshooting helps inform a user not only of a problem but also at what point in
10 the network connecting process a failure has occurred.

11 12 Exemplary Computing Device

13 **Fig. 8** shows an exemplary computer 109 suitable as an environment for
14 practicing aspects of the subject matter. The components of computer 109 may
15 include, but are not limited to, a processing unit 820, a system memory 830, and a
16 system bus 821 that couples various system components including the system
17 memory 830 to the processing unit 820. The system bus 821 may be any of
18 several types of bus structures including a memory bus or memory controller, a
19 peripheral bus, and a local bus using any of a variety of bus architectures. By way
20 of example, and not limitation, such architectures include Industry Standard
21 Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA
22 (EISA) bus, Video Electronics Standards Association (VESA) local bus, and
23 Peripheral Component Interconnect (PCI) bus also known as the Mezzanine bus.

24 Exemplary computer 109 typically includes a variety of computer-readable
25 media. Computer-readable media can be any available media that can be accessed

1 by computer 109 and includes both volatile and nonvolatile media, removable and
2 non-removable media. By way of example, and not limitation, computer-readable
3 media may comprise computer storage media and communication media.
4 Computer storage media include volatile and nonvolatile, removable and non-
5 removable media implemented in any method or technology for storage of
6 information such as computer-readable instructions, data structures, program
7 modules, or other data. Computer storage media includes, but is not limited to,
8 RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM,
9 digital versatile disks (DVD) or other optical disk storage, magnetic cassettes,
10 magnetic tape, magnetic disk storage or other magnetic storage devices, or any
11 other medium which can be used to store the desired information and which can be
12 accessed by computer 109. Communication media typically embodies computer-
13 readable instructions, data structures, program modules or other data in a
14 modulated data signal such as a carrier wave or other transport mechanism and
15 includes any information delivery media. The term "modulated data signal"
16 means a signal that has one or more of its characteristics set or changed in such a
17 manner as to encode information in the signal. By way of example, and not
18 limitation, communication media includes wired media such as a wired network or
19 direct-wired connection and wireless media such as acoustic, RF, infrared and
20 other wireless media. Combinations of any of the above should also be included
21 within the scope of computer readable media.

22 The system memory 830 includes computer storage media in the form of
23 volatile and/or nonvolatile memory such as read only memory (ROM) 831 and
24 random access memory (RAM) 832. A basic input/output system 833 (BIOS),
25 containing the basic routines that help to transfer information between elements

1 within computer 109, such as during start-up, is typically stored in ROM 831.
2 RAM 832 typically contains data and/or program modules that are immediately
3 accessible to and/or presently being operated on by processing unit 820. By way
4 of example, and not limitation, Fig. 8 illustrates operating system 834, application
5 programs 835, other program modules 836, and program data 837. Although an
6 exemplary NCAT 200 is depicted as software in random access memory 832,
7 other implementations of an exemplary NCAT 200 can be hardware or
8 combinations of software and hardware.

9 The exemplary computer 109 may also include other removable/non-
10 removable, volatile/nonvolatile computer storage media. By way of example only,
11 Fig. 8 illustrates a hard disk drive 841 that reads from or writes to non-removable,
12 nonvolatile magnetic media, a magnetic disk drive 851 that reads from or writes to
13 a removable, nonvolatile magnetic disk 852, and an optical disk drive 855 that
14 reads from or writes to a removable, nonvolatile optical disk 856 such as a CD
15 ROM or other optical media. Other removable/non-removable,
16 volatile/nonvolatile computer storage media that can be used in the exemplary
17 operating environment include, but are not limited to, magnetic tape cassettes,
18 flash memory cards, digital versatile disks, digital video tape, solid state RAM,
19 solid state ROM, and the like. The hard disk drive 841 is typically connected to
20 the system bus 821 through a non-removable memory interface such as interface
21 840, and magnetic disk drive 851 and optical disk drive 855 are typically
22 connected to the system bus 821 by a removable memory interface such as
23 interface 850.

24 The drives and their associated computer storage media discussed above
25 and illustrated in Fig. 8 provide storage of computer-readable instructions, data

1 structures, program modules, and other data for computer 109. In Fig. 8, for
2 example, hard disk drive 841 is illustrated as storing operating system 844,
3 application programs 845, other program modules 846, and program data 847.
4 Note that these components can either be the same as or different from operating
5 system 834, application programs 835, other program modules 836, and program
6 data 837. Operating system 844, application programs 845, other program
7 modules 846, and program data 847 are given different numbers here to illustrate
8 that, at a minimum, they are different copies. A user may enter commands and
9 information into the exemplary computer 109 through input devices such as a
10 keyboard 854 and pointing device 861, commonly referred to as a mouse,
11 trackball, or touch pad. Other input devices (not shown) may include a
12 microphone, joystick, game pad, satellite dish, scanner, or the like. These and
13 other input devices are often connected to the processing unit 820 through a user
14 input interface 860 that is coupled to the system bus, but may be connected by
15 other interface and bus structures, such as a parallel port, game port, or a universal
16 serial bus (USB). A monitor 898 or other type of display device is also connected
17 to the system bus 821 via an interface, such as a video interface 890. In addition
18 to the monitor 898, computers may also include other peripheral output devices
19 such as speakers 897 and printer 896, which may be connected through an output
20 peripheral interface 895.

21 The exemplary computer 109 may operate in a networked environment
22 using logical connections to one or more remote computers, such as a remote
23 computer 880. The remote computer 880 may be a personal computer, a server, a
24 router, a network PC, a peer device or other common network node, and typically
25 includes many or all of the elements described above relative to computer 109,

1 although only a memory storage device 881 has been illustrated in Fig. 8. The
2 logical connections depicted in Fig. 8 include a local area network (LAN) 871 and
3 a wide area network (WAN) 873, but may also include other networks. Such
4 networking environments are commonplace in offices, enterprise-wide computer
5 networks, intranets, and the Internet.

6 When used in a LAN networking environment, the exemplary computer
7 109 is connected to the LAN 871 through a network interface or adapter 870.
8 When used in a WAN networking environment, the exemplary computer 109
9 typically includes a modem 872 or other means for establishing communications
10 over the WAN 873, such as the Internet. The modem 872, which may be internal
11 or external, may be connected to the system bus 821 via the user input interface
12 860, or other appropriate mechanism. In a networked environment, program
13 modules depicted relative to the exemplary computer 109, or portions thereof, may
14 be stored in the remote memory storage device. By way of example, and not
15 limitation, Fig. 8 illustrates remote application programs 885 as residing on
16 memory device 881. It will be appreciated that the network connections shown
17 are exemplary and other means of establishing a communications link between the
18 computers may be used.

21 CONCLUSION

22 The foregoing describes an exemplary network connection agent and
23 troubleshooter (NCAT) and related methods for connecting a device to a network
24 and/or network service. The subject matter described above can be implemented
25 in hardware, in software, or in both hardware and software. In certain

1 implementations, the exemplary NCAT and related methods may be described in
2 the general context of computer-executable instructions, such as program modules,
3 being executed by a computer. Generally, program modules include routines,
4 programs, objects, components, data structures, etc. that perform particular tasks
5 or implement particular abstract data types. The subject matter can also be
6 practiced in distributed communications environments where tasks are performed
7 over wireless communication by remote processing devices that are linked through
8 a communications network. In a wireless network, program modules may be
9 located in both local and remote communications device storage media including
10 memory storage devices.